A Reference Model for Data Mapping tools

Draft Update August 2013

1. Introduction

This is a draft of a reference model for mapping tools. The intension is to address a comprehensive and sustainable functionality in a scenario of information providers and aggregators, including the long-term maintenance of resources. We assume a distribution of responsibilities, in which the information provider curates his/her resources and provides in regular intervals updates, whereas the aggregator is responsible for the homogeneous access to the integrated data and the resolution co-references (multiple URIs for the same thing) across all provided data. In the course of the transformation of resources to the target system, some kinds of quality control can be done which the provider has no means to do (see also the services provided by OCLC). Therefore the information provider receives and benefits from data cleaning information produced by the aggregator. The challenge is to define a modular architecture of as many components as possible that can independently be developed and optimized with minimal interfaces between them, without hindering integrated UI development for the different user roles involved.

The first part of this model is a sort of requirements specification, which breaks down in the usual way into a definition of the involved user roles, the primary kinds of data the users aims at handling and the complete definition of the processes users carry out to manage these data.

2. Process Model

In the following, we specify user roles and data objects in order to have a vocabulary to define user processes. Of course, these things are interdependent, so a certain redundancy is inevitable. We do not regard IT processes as self-contained and opposed to user processes, but IT processes are regarded as being part of the user processes replacing manual work. Once such a model has been defined, it allows for a dynamic definition of which user processes are replaced or assisted by IT processes, and to justify IT processes in terms of their utility for the functions users are ultimately interested in.

2.1 User Roles

Primary user roles are:

Provider: We call providers in this model the maintainers of *Local information systems*. In this model, we may also call them simply *source systems*. Following CIDOC CRM v5.0, “These are either collection management systems or content management systems that constitute institutional memories and are maintained by an institution. They are used for primary data entry, i.e. a relevant part of the information, be it data or metadata, is primary information in digital form that fulfils institutional needs”. In practice and even more general, these are individual museums, archives, libraries, sites and monument records, academic institutes, private research societies etc., represented by their curators, IT referents or researchers. Providers *ultimately* have the knowledge about

the meaning of their data in the real world (if anybody has it), or know who knows, or know how to verify it.

Aggregator: We call aggregators in this model the maintainers of *Integrated access systems*. In this model, we may also call them simply *target systems*. Following CIDOC CRM v5.0, "These provide an homogeneous access layer to multiple local systems. The information they manage resides primarily on local systems." Aggregators will maintain a form of business agreements with providers to send data from local systems to the aggregators' systems, primarily metadata. In this model, we are not interested in aggregators doing harvesting without any business interaction with the provider. The model will be useful for such scenarios in a trivial way we do not explicitly describe. Aggregators have no direct knowledge about the meaning of the data they aggregate.

Secondary user roles are experts whose knowledge or services contributes to the mapping process:

Source Schema Expert: The curator, researcher or data manager of the local system who is responsible for the semantically correct data entry into the local system, i.e., the one who knows how fields, tables or elements in the schema correspond to the reality described by them following *local* use and practice.

Target Schema Experts: The expert(s) for the semantics of the schema employed by the aggregator ("integration model"). It is very likely that the aggregator uses a more widely known standard schema. Typically but not exclusively, we talk in this model about the CIDOC CRM and extensions of it. Therefore the target schema experts may not need to have intimate knowledge of the aggregator's context. Moreover, there does not **yet** exist an established practice of a curator role at the aggregator side. Nevertheless, the requirement for semantic consistency in practice forces such a role to exist de facto, often a user team including some provider representatives. Therefore the target schema experts should include or be in close contact with a sort of curators of the integrated access system to fulfill this role.

URI Expert: The expert of the aggregator, normally an IT specialist, who is responsible for maintaining the referential integrity of the (meta)data in the integrated access system and who knows how to generate from provider data valid URIs for the integrated access system.

Source Terminology Expert: The curator, maintainer or other expert of one of the terminologies which the provider use as reference in the local system. If the terminology is provided by a third party, such as the Getty Research Institute, there may exist independent experts trained in this terminology. If it is local or even uncontrolled, it is typically the curators or other local data managers who know the meaning of the local terms.

Target Terminology Expert: The curator, maintainer or other expert of one of the terminologies which the aggregator uses as reference in the integrated access system. Aggregator normally want to avoid to engage in the terminology maintenance business. They will rather use and refer to third party terminology, or take over provider term lists.

Mapping manager: The actor responsible for the maintenance of the data transformation process from the provider format to the aggregator format. This role may split into a semantic and a technical part, and may be regarded as either an aggregator

task, a provider task or a user consortium's task. To our opinion, the mapping technology this model aims at should enable a scalable management of the data transformation process by the aggregator.

2.2 Data objects

We distinguish the following primary data objects:

Source systems: I.e., *local information systems* in the sense of the CIDOC CRM v5.0 ("These are either collection management systems or content management systems that constitute institutional memories and are maintained by an institution") from which (meta)data are sent on a regular base or in a single action to some aggregator. We are interested here in their typical role relative to the processes we describe, regardless if they also may play some target system role to a third party.

Target system: i.e., *integrated access systems* in the sense of the CIDOC CRM v5.0 ("These provide an homogeneous access layer to multiple local systems. The information they manage resides primarily on local systems"), to which (meta)data are sent on a regular base or in a single action by several providers. We are interested here in their typical role relative to the processes we describe, regardless if they also may play some source system role to a third party.

Terminologies: Controlled vocabularies of terms that appear as individual data *values* in the source or target systems. We do **never** use the term "vocabulary" for metadata schemata. Terminologies may be flat list of words or be described and organized in more elaborate structures as so-called "thesauri" or "knowledge organization systems", the most popular format now being SKOS. Here, we do **not** use the term "ontology", even if the terminology may qualify as such, as long as its use in this context is to provide *data values*.

Content objects: Individual files or information units with an internal structure that is **not** described in terms of schema elements of the source or target systems. These are typically images or text documents, and searched by content retrieval indices such as keyword search, rather than by associative queries. They are described as objects by metadata records which are searched by associative queries. Important in this context is not the actual structure of an information unit to be qualified as a content object, but the way it is treated in the information system (sometimes also called "blobs"). Many aggregators do not collect content object but only link them back to the provider system.

Metadata records: Information units with an internal structure that **is** described in terms of schema elements of the source or target systems. In our context, these are often data records describing content object (therefore the term "metadata"), but bad analogy brought the term also into use for data describing physical objects and other historical contexts. Therefore we define it here by the way it is treated in the information system, and not as "data about data". The metadata records are the common target of submission to aggregators and therefore of transformation from the source to the target schema.

Secondary data objects are those that support the mapping processes:

Source schema definitions: data dictionaries, XML schemata, RDFS/OWL files etc. describing the data structures that are managed and can be searched by associative queries in the source or systems.

Integration Model: The definition of the schema of the target system, now mostly an RDFS/OWL knowledge representation model ("ontology").

Other kinds of data objects which are *part of* this reference model in the sense of products or interface definitions of the components it foresee, such as schema matching definitions etc. (see below)

2.3 Mapping processes

2.3.1 Overview

This reference model aims at identifying, supporting or managing the processes needed to be executed or maintained when a provider and aggregator agree

(1) to ***transfer data*** from the provider to the aggregator,

(2) to ***transform*** their format to the (homogeneous) format of the aggregator,

(3) to ***curate*** the semantic consistency of source and target data and the global referential integrity and

(4) to maintain the transferred data ***up-to-date*** with whatever relevant changes occur in the source and target systems and the employed terminologies.

Note that experienced aggregators keep the original data from the provider, so that they can reexecute the data transformation process without asking for resubmission.

Figure 1: Mapping Processes diagram

At a first level, this breaks down into the following independent processes:

(a) Management of which data will be delivered and processed at what time, including updates.

(b) The mapping definition, i.e., specification of the parameters for the data transformation process, such that complete sets of data records can automatically be transformed, manual exception processing notwithstanding. This includes harmonization between multiple providers.

(c) The actual transfer of data until a first consistent state is achieved. This includes transformation of sets of data records submitted to the aggregator, the necessary exception processing of irregular input data between provider and aggregator, ingestion of the transformed records into target system and initial referential integrity processing possibly on both sides.

(d) Referential integrity processing at the aggregator side out of the context of a particular data submission, which is not our concern in this model.

(e) Change detection and update processing to restore ability of data transformation and semantic consistency, which comprises changes in the source target records, in the source or target terminologies, in the source or target schemata, in the target URI policy and in the good practice of interpretation of source and target schema in the mapping definition.

Only if these processes are sustained, an aggregator can provide valid and consistently integrated data in long terms, and thereby deliver the full added value of an aggregation service as a resource

for professional and private research, which ultimately justifies its existence. We observe that absolutely none of the dozens or hundreds of mapping tools and frameworks created in numerous projects has ever systematically addressed this comprehensive scenario.

2.3.2 Analytical Representation

2.3.2.1 Data Delivery Management

Data Delivery Management deals with which data will be delivered and processed at what time, including updates. A *Mapping Manager* may be responsible for this task, possibly with a custodial participation or supervision on provider and aggregator side.

Here we do not further analyze the subprocesses which do not affect the mapping itself. In general, IT- support will be given by log-files of delivered content, facilities to query (last) changes of source system records or other data requiring resubmission and queries to support selection criteria in the source systems. Also, queries in the target system may be used to reveal semantic needs in the composition of the aggregation and to derive requests for certain materials from providers.

There is however a set of characteristic changes in the provider – aggregator environment that affect the mapping and may require

redefinition of the mapping

reexecuting the transformation of records already submitted to the aggregator and updating the transformed records in the target system

resubmission of records from the source system.

The mapping manager must monitor such changes and initiate respective actions.

2.3.2.2 Mapping Definition

Mapping definition breaks down into

Syntax normalization

Schema matching,

URI generation specification

Terminology mapping

The *Mapping Manager* may be responsible for issuing and coordinating these tasks.

2.3.2.2.2 Syntax Normalization

Some provider systems do not employ the data or database formats of RDBMS, MSExcel spreadsheets, XML or RDF/OWL which are now standard, but may use tables in text documents or other formats. The latter do generally not impose formal control on the syntax corresponding to the intended semantics. Even standard formats allow for the use of free text fields where users may insert their own syntactic inventions without a formal control, such as punctuation marks, italics, parentheses and others. XML further allows for semi-structured data and the use of elements not appearing in a schema declaration. Local identifiers may have their own syntactic structure, such as inventory numbers, addresses, bibliographic references, date and time etc.

Automated data transformation, i.e. transformation of data from one schema to another without loss of meaning or with controlled loss of meaning by a deterministic algorithm based on a mapping definition is only possible if the part of the data to be transformed (not the free text descriptions) is completely structured. Therefore, whatever data is to be mapped and transformed has to be completely structured and each structural element must have a clearly described meaning.

Since data transformation tools can only deal with a limited set of standard data structures, any non-standard forms must be converted to a standard one, i.e., RDBMS, XML, RDF OWL or at least spreadsheets. In particular XML is so powerful that virtually any user invented syntax can be normalized to XML. Even XML records without explicit schema can automatically be analyzed for the actually used tags and syntax.

For local identifier formats, such as dates or inventory numbers, it may not be worthwhile to normalize their internal structure to XML prior to the mapping, but rather use right away custom code for the URI generation (see section 2.3.2.2.3).

The syntax normalization can be done by an *IT expert*, possibly the same one dealing with *URI generation*, in collaboration with the *source schema expert*. Local syntax rules can be so complicated or even deterministic that it is often more effective to use a set of custom filtering routines, resolving one structural feature at a time, and verifying it with the *source schema expert*. For instance, if italics are used to tag a particular kind of field, it is better to convert first all italics to XML tags.

This process will reveal data that are inconsistent with the user's own intentions, and need to be mended. There will be a residual of cases so complicated to describe by rules, that manual rewriting is more effective. This is not a bottle neck, or a reason not to proceed. The important thing is to drastically reduce the number of records to be checked and treated manually. Therefore it is equally important to find diagnostic rules for inconsistent cases as it is to resolve those that can be formally described. If within those formally described inconsistent ones "slip through" undetected, all records have to be reviewed manually. That would indeed become a bottleneck.

After syntax normalization, we expect all data structures relevant for the transformation to be in a standard form. Note that in this step **NO approximation** of the target schema semantics should be attempted. Rather, it must be an exact representation of the user's conceptualization. Diego Calvanese et al. [XXXX] talk about a "source model".

2.3.2.2.2 Schema Matching

Source schema experts and a *target schema expert* (e.g., CIDOC CRM) define a schema matching, which is documented in a ***schema matching definition*** file. This file must be human and machine readable and is the ultimate communication means on the semantic correctness of the mapping. The collaboration between these *experts* must be well organized and is the bottle-neck of the mapping process.

In order to do so, all source schema elements must be well understood and mapped to target schema paths. Therefore also the target schema must be well understood. Both tasks need two

independent tools to visualize source and target schema and source metadata records. Adequate navigation and viewing techniques must allow for overviews and understanding of details.

The matching process must lead the user through all source elements in order to make a mapping decision. This may be supported by tools *suggesting mappings* (automated mapping). The automated mapping tools should recalculate their proposals with each new mapping decision by some user. They should make use of "*mapping memories*" of analogous cases collected from the user community. An aggregator may maintain *mapping guidelines* together with provider and user consortia.

The matching may need to interpret source or target terminologies in order to re-solve data dependent mappings. For instance, a field "object type = Vase" may indicate a "Physical Object", and "object type = Image" an "Information Object". Such difference will lead to a thorough reinterpretation of most other fields describing such an object. Since the 1980ties it is known that schema semantics and categorical data ("terms") cannot be separated. In the *schema matching definition*, we generally foresee *mapping conditions* that a term in the source record is equal to or unequal to a constant, or a narrower term of a constant. This may be expressed in terms of *source or target terminology*. In order to resolve these specifications at *record transformation time*, partial *terminology mappings* of source and target terminology must exist.

All related tools should take into account the need for *incremental mappings* after source or *target schema definition* up-dates, *terminology* updates and *mapping guideline* updates and guide the user through relevant changes.

Some source data or source schema elements may not allow for matching decisions and need improvement by the provider. It must be possible to define filters for these data that run before and at transformation-time and feed them back to the provider.

2.3.2.2.3 URI Generation Specification

After the *schema matching* process, the URI generation policies for each instance of a *target schema* class referred to in the matching must be defined, such as for persons, objects, events, places, and formats of time. This is typically a task the *source schema expert* has **not** interest in or knowledge for. It depends on the aggregators policies for *co-reference resolution* in the *target system*. It is a task for an *URI expert.* The URI generation policies can be introduced in an abstract form as rules or references to code signatures implementing specific rules, complementing the *schema matching definition* file into a *mapping definition* file.

Some URI generation policies may include *look-up of on-line resources ("authority records")*about persons or places at transformation time. Such look-up may fail, and adequate exception handling must be foreseen.

The execution of URI generation rules may reveal "dirty data" of the provider at transformation time, or before. "Dirty data filters" must be foreseen in the generation rules. The source metadata records may be analyzed before transformation time for such cases. Providers must be informed about "dirty data" cases, and given the possibility to run an organized, sustainable process to improve the source data.

It may be possible to define preliminary workarounds to maintain the submission process, i.e. characteristic data patterns replacing dirty data that can later be recognized and updated at aggregator side without resubmission. Otherwise, "dirty records" are held back until they are updated.

Changes of URI policies of the aggregator may result in the need to update the URI generation rules. Changes of naming and identifier policies at the provider side may also make a redefinition of URI generation rules necessary. It must be possible to do that without affecting the *schema matching definition* file.

2.3.2.2.4 Terminology Mapping

Terminology mapping can be a huge task. Providers may use anything from intuitive lists of uncontrolled terms up to highly structured third party thesauri. However, most of the provider terminology is very specialized and more important as information element when metadata records are displayed than as search term in the target system. As long as the terms are in the same natural language, most terms can just be copied from the source records into the transformed target records. If they are in other languages, aggregators may choose to translate terms, possibly preserving also the provider terms. It may be useful to associate provider terms with broader terms of some standard terminology the aggregator employs as search terms. Since all this can happen even after metadata record transformation, it does not affect the mapping process itself.

In this model, we are only interested in the consistency of the mapping process when the choice of a target class or property depends on a term. For that sake, we can extract from the schema matching definition the terms appearing in mapping conditions. We distinguish two cases:

(a) equality/ inequality condition: These terms (constants) must be taken from the provider terminology and no action is needed.

(b) broader term condition:

1) If the constant term is given in the provider terminology, the narrower term hierarchy for each constant term is used if it exists, otherwise it must be "invented". The latter is one case of terminology mapping.

2) If the constant term is given in the aggregator terminology, for each constant term the narrower terms in the source terminology must be identified. This is the second case of terminology mapping.

In any case, the aggregator terminology should have a thesaurus structure, albeit a small vocabulary of high-level terms. Sometimes it may be more effective to merge provider terms with aggregator terms, i.e., replace equivalent terms and insert all other provider terms as narrower terms of aggregator terms.

In this case, the provider terminology should be *replaced* by the updated aggregator terminology ***before transforming*** the respective records and in the schema matching definition file. This will allow for better controlling the mutual consistency of mappings between different providers.

The terminology mapping may reveal "dirty data" of the provider. "Dirty data filters" must be foreseen for terminology. The source metadata records should be analyzed before transformation time for such cases. Providers must be informed about "dirty data" cases, and given the possibility to run an organized, sustainable process to improve the source data.

It may be possible to define preliminary workarounds to maintain the submission process, i.e. characteristic data patterns replacing dirty data that can later be recognized and updated at aggregator side without resubmission. Otherwise, "dirty records" are held back until they are updated.

Mapping identifiers of persons and places to those used at the aggregator system is in general ineffective due to the large number of such identifiers, most of which are only known at local level. It is more effective to update the provider with identifiers (URIs) of persons and places referred to by more than one provider (or third party authority, such as viaf.org)

After these steps, metadata records are ready for transformation.

2.3.2.3 Metadata Transfer

2.3.2.3.1 Submission

In order to transform source metadata, they must be either submitted to the aggregator or to the ***mapping manager*** or they are harvested by a protocol like the OAI-PMH.

The submitted metadata records must be indentified with a unique identifier, checksum and modify date-time. The submission management must be able to recognize any change of a source record by the metadata record metadata.

2.3.2.3.2 Transformation

The transformation process itself may run *completely automatically*. If it encounters dirty data, it may either sort out dirty records and **send them back** to the provider for processing. Otherwise, if possible and given sufficient, trusted expertise, *experts* under the control of the *mapping manager* may correct some of them manually in an *interactive process*. The result is a set of valid target records. An automated translation for source terms using a terminology map may follow.

The URI generation algorithm of the automatic data transformation process may employ an initial instance matching process with the target system in order to reuse already existing URIs in the target system. This also holds for third party authority systems with URIs the aggregator uses as reference (such as viaf.org). In any case, such a matching should be reported back to the provider, and the provider should better update his records with such URIs (possibly in addition to his local ones).

2.3.2.3.3 Ingest and Storage

The transformed records will be ingested into the target system.

An aggregator should store all source metadata records which are going to be transformed, or which are transformed and have been ingested to the target system. The aggregator must preserve the link to the identity and version of the source record it is derived from. Some aggregators return also fitting source records in query results (e.g., the German Digital Library). Source records may be syntactically normalized for that purpose.

If a new version of a transformed source record is ingested in the target system, the target record representing the previous version must be deleted in the target system if it has the form of a semantic network (without a “context” or “named graph” or “quad store” mechanism, this causes a “data warehouse update problem” at target system in the form of a Triple Store,).

The aggregator may decide to delete copies of updated source records. Providers should better keep the old versions. If providers have no means to do so (e.g., in RDBMS), aggregators may take over the preservation of old versions as a service.

2.3.2.4 Co-reference Resolution

The aggregator may continue at any time with referential integrity processing, i.e., resolving that multiple identifiers denote the same real world thing or object of discourse (co-reference resolution). This is a process in its own right we are not concerned about here (yet). The main goals are to ensure referential integrity, which is the heart of information integration, and to reduce the number of URIs in use for the same thing. It requires its own dialogue between provider, aggregator and third-party authority managers. Since the aggregator collects more comprehensive knowledge than the providers, it is a natural role of the aggregator. One may regard that the only genuine knowledge of the aggregator is the co-reference knowledge.

2.3.2.5 Change Detection and Update Processing

The mapping manager must monitor all changes that may affect the consistency of provider and aggregator data.

Those changes are:

1. new source records
2. source record updates (new versions)
3. source schema changes
4. provider changes identifier policy (for people, objects, events, places, time) and updates his records
5. provider changes terminology data (terms or authority) and updates his records
6. provider changes terminology structure (broader term links etc.)
7. target schema changes
8. aggregator changes URI policy
9. aggregator changes terminology (terms or authority)
10. aggregator or user consortium changes mapping guidelines
11. Source-target terminology mapping changes

The changes of kind 11 requires retransformation and re-ingestion of all (stored) source records already transferred which refer to the respective terms.

The changes of kind 6 and 9 require redoing the terminology mapping (2.3.2.2.4), retransformation and re-ingestion of all (stored) source records already transferred which refer to the respective terms.

Changes of kind 8 require updating the URI generation specification in the mapping file, retransformation and re-ingestion of all (stored) source records already transferred which refer to the respective terms.

Changes of kind 7 and 10 require updating the schema matching definition, retransformation and re-ingestion of all (stored) source records already transferred which refer to the respective terms.

The changes of kind nr. 1 and 2 require running the complete metadata transfer (2.3.2.3) with the changed or new source records but using the existing mapping definition.

Changes of kind 3 require updating the schema matching definition in the mapping file, resubmission of all source records affected, transformation and ingestion, replacing the target records transformed from the previous version of these source records.

Changes of kind 4 require updating the URI generation specification in the mapping file, resubmission of all source records affected, transformation and ingestion, replacing the target records transformed from the previous version of these source records.

Changes of kind 5 require updating the terminology mapping, resubmission of all source records affected, transformation and ingestion, replacing the target records transformed from the previous version of these source records.

3. Component Design